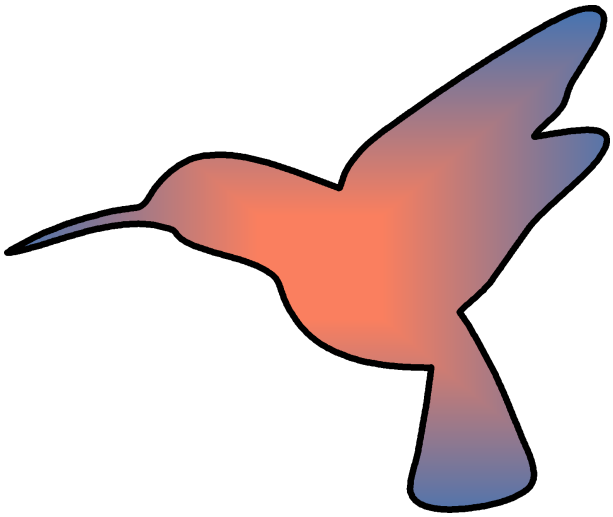
User Manual

Colibri

Inertial Motion Tracker



(Subject to technical modifications)

Copyright © 2011

Augmented Vision Group of the

German Research Center for Artificial Intelligence GmbH

(http://www.dfki.de)

TRIVISIO

(http://www.trivisio.com)

Contents

1 Introduction **3**

2 Installation **4**
2.1 Windows 4
2.2 Linux 6

3 GUI **7**
3.1 Starting the application 7
3.2 The GUI 8
3.3 Calibration parameters 11
3.4 Boresighting 12
3.5 Jitter Reduction 13
3.6 Additional functionality 13
3.6.1 Disable drawing of textures 13
3.6.2 Additional COM ports to scan 14
3.6.3 Sensor Diagnosis 15
3.6.4 Firmware Upgrade 16

4 SDK **18**
4.1 The `test.c` example 18
4.2 The `orientation.cpp` example 21
4.3 Simple Import mechanism for data profiling 21

Chapter 1

Introduction

Thank you for purchasing the Trivisio Colibri and using its software development kit (SDK).

This user's manual covers the software available for the Colibri Tracker. Chapter 2 is an installation guide for the software. The graphical user interface (GUI) provided with the sensor is described in Chapter 3. It covers the functionality of the GUI and does also gives some hint on how to get the most out of the Colibri Tracker. Chapter 4 describes two examples of using the SDK's API. The code of one of the examples is studied in detail.

Chapter 2

Installation

This chapter provides the installation guides for Microsoft Windows and Linux.

If you do not have an installer yet, please download the installation file from the Trivisio GmbH website: http://www.trivisio.com. Installers for Window and Linux (32 and 64 bit) can be freely downloaded from the *Support — Software* section (http://www.trivisio.com/index.php/support/software).

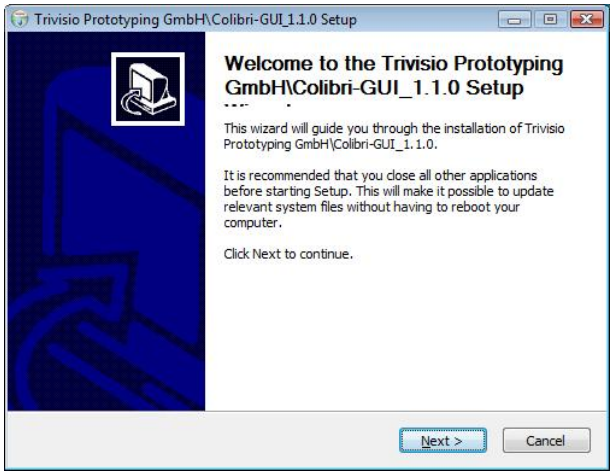
Download the installer for your system, and continue to the section covering the installation guidelines for your operating system.

2.1 Windows

Run the installer, the following images will then guide you through the installation procedure.

Step 1: Welcome Screen

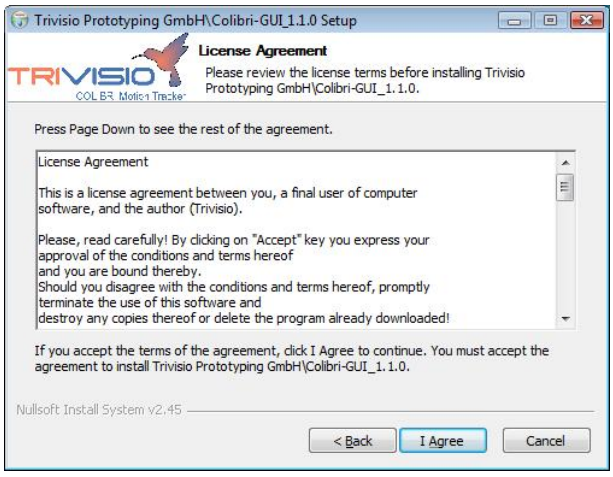
This is the welcome screen of the installer. Press next to start the installation or cancel to abort.


Trivisio Prototyping GmbH\Colibri-GUI_1.1.0 Setup
Welcome to the Trivisio Prototyping GmbH\Colibri-GUI_1.1.0 Setup
This wizard will guide you through the installation of Trivisio Prototyping GmbH\Colibri-GUI_1.1.0.
It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.
Click Next to continue.
Next >
Cancel

Step 2: EULA

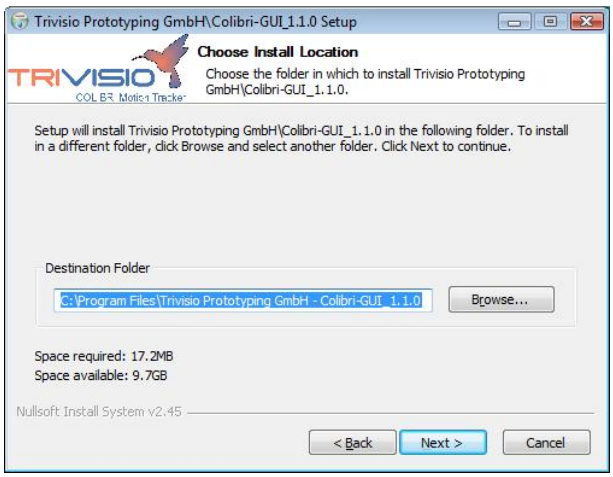
The next dialog is the End User License Agreement. Please read it carefully.

If you agree to the terms and conditions, please press the “I Agree” button to continue the installation procedure.


Trivisio Prototyping GmbH\Colibri-GUI_1.1.0 Setup
TRIVISIO
License Agreement
Please review the license terms before installing Trivisio Prototyping GmbH\Colibri-GUI_1.1.0.
Press Page Down to see the rest of the agreement.
License Agreement
This is a license agreement between you, a final user of computer software, and the author (Trivisio).
Please, read carefully! By clicking on "Accept" key you express your approval of the conditions and terms hereof and you are bound thereby.
Should you disagree with the conditions and terms hereof, promptly terminate the use of this software and destroy any copies thereof or delete the program already downloaded!
If you accept the terms of the agreement, click I Agree to continue. You must accept the agreement to install Trivisio Prototyping GmbH\Colibri-GUI_1.1.0.
Nullsoft Install System v2.45
< Back
I Agree
Cancel

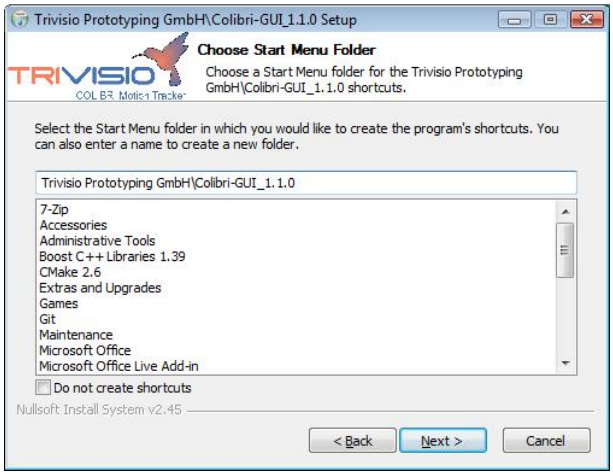
Step 3: Installation path

In this dialog the user is able to choose or select the desired installation path.


Trivisio Prototyping GmbH\Colibri-GUI_1.1.0 Setup
TRIVISIO
Choose Install Location
Choose the folder in which to install Trivisio Prototyping GmbH\Colibri-GUI_1.1.0.
Setup will install Trivisio Prototyping GmbH\Colibri-GUI_1.1.0 in the following folder. To install in a different folder, click Browse and select another folder. Click Next to continue.
Destination Folder
C:\Program Files\Trivisio Prototyping GmbH - Colibri-GUI_1.1.0
Browse...
Space required: 17.2MB
Space available: 9.7GB
Nullsoft Install System v2.45
< Back
Next >
Cancel

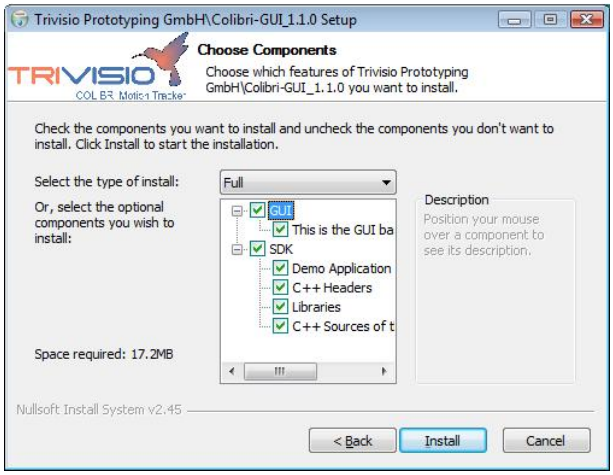
Step 4: Start menu folder

If you want to have a different name for your start menu folder, please modify the path to your personal need.


Trivisio Prototyping GmbH\Colibri-GUI_1.1.0 Setup
Choose Start Menu Folder
Choose a Start Menu folder for the Trivisio Prototyping GmbH\Colibri-GUI_1.1.0 shortcuts.
Select the Start Menu folder in which you would like to create the program's shortcuts. You can also enter a name to create a new folder.
Trivisio Prototyping GmbH\Colibri-GUI_1.1.0
7-Zip
Accessories
Administrative Tools
Boost C++ Libraries 1.39
CMake 2.6
Extras and Upgrades
Games
Git
Maintenance
Microsoft Office
Microsoft Office Live Add-in
Do not create shortcuts
Nullsoft Install System v2.45
< Back
Next >
Cancel

Step 5: Component installation

The next dialog can be used to select different installation types. The default installation type is **FULL**.


Trivisio Prototyping GmbH\Colibri-GUI_1.1.0 Setup
Choose Components
Choose which features of Trivisio Prototyping GmbH\Colibri-GUI_1.1.0 you want to install.
Check the components you want to install and uncheck the components you don't want to install. Click Install to start the installation.
Select the type of install:
Full
Or, select the optional components you wish to install:
GUI
This is the GUI ba
SDK
Demo Application
C++ Headers
Libraries
C++ Sources of t
Description
Position your mouse over a component to see its description.
Space required: 17.2MB
Nullsoft Install System v2.45
< Back
Install
Cancel

The different options are: Full, Developer, GUI, and Custom. The **Full** installation comprise the **Developer** and **GUI** installations.

After this dialog the installation is executed and you are able to use the applications and develop your own software based on the SDK.

2.2 Linux

Simply run the installer, accept the EULA, and follow the instruction presented.

Chapter 3

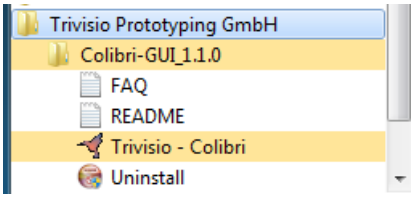
GUI

This section provides an overview about the graphical user interface, which can be used to view the connected sensors and their values.

3.1 Starting the application

Using Microsoft Windows

You can start the application either by clicking the **Trivisio – Colibri** icon on your desktop or by selecting it from the start menu. The default location is: **Trivisio – Colibri X.X.X** (see below) and execute the **Trivisio – Colibri**.

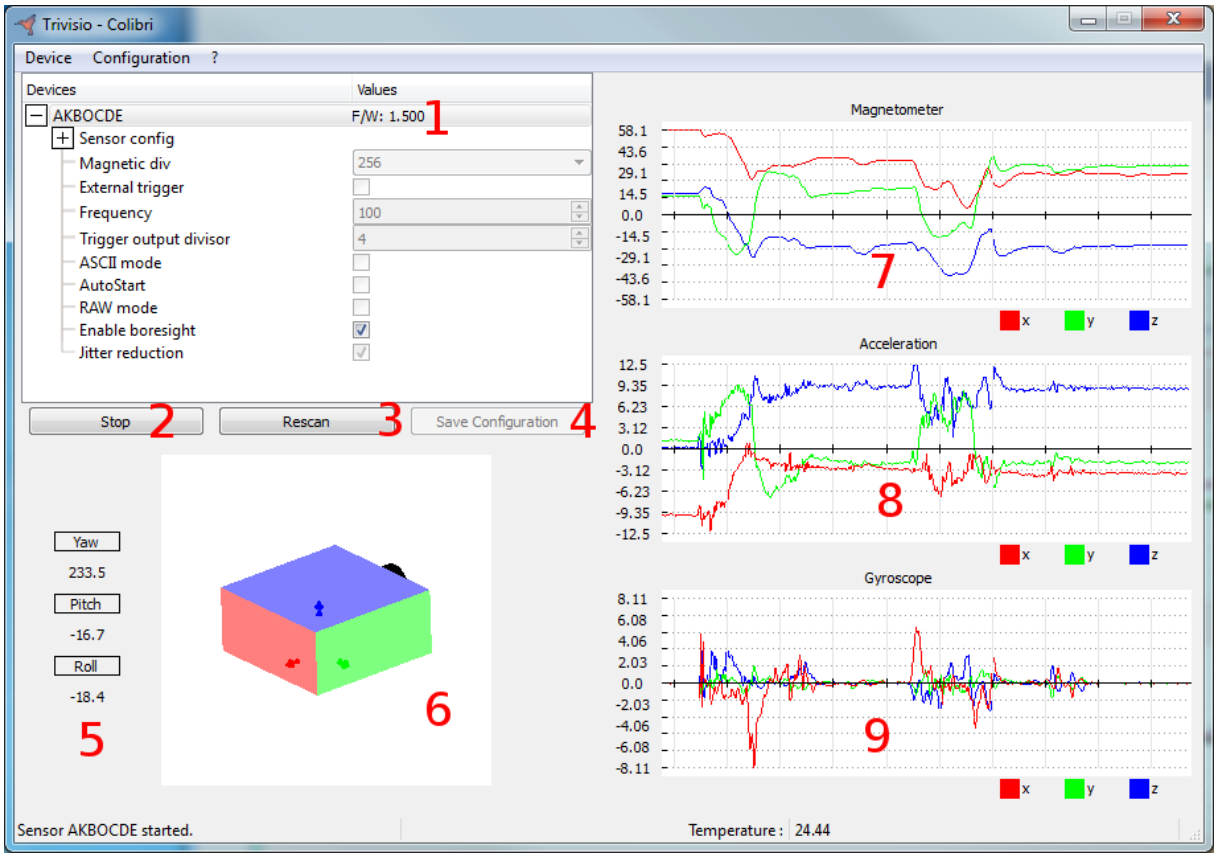


Using Linux

Execute the `TrivisioGUI` executable, available in the `bin` directory of the installation.

3.2 The GUI

This is a snapshot of the GUI application.



Based on this image, the functionality will be described briefly below:

1. The list of Colibri sensors found connected to the system.

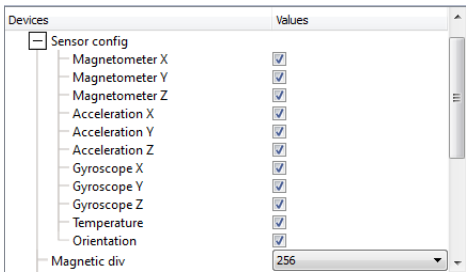
   Detailed information about each sensor can be viewed and changed by expanding the tree view by pressing the "plus" symbol. There might be sensors marked with red. These are equipped with an outdated firmware version, which must be updated before the sensor can be accessed or used. See Section 3.6.4. The sensors are identified by their serial number and firmware version.

   When a sensor is not running, it is possible to change its settings. The boresighting setting, further described in Section 3.4, is an exception to this rule as it can also be changed when the sensor is running.

   By expanding the **Sensor Config** (see figure below), it is possible to see what individual sensor elements are enabled in the Colibri. The user can enable the individual *magnetometers*, *accelerometers*, and *gyroscopes*, and also the built in temperature sensor. The *Orientation* sensor turns on the Colibri's virtual orientation sensor. When using the orientation sensor, all accelerometers, gyroscopes and magnetometers will automatically be turned on and cannot be deactivated before the orientation is deactivated.

   The *Magnetic div* determines the sensitivity of the magnetometers. A higher value improves the magnetic measurements, but does also take more time to measure.

RAW	ASCII	Enabled displays
off	off	Sensor and graphs
on	off	Only graphs
off	on	No output
on	on	No output



Sensors that can be enable.

Hence, if you experience problems with too low update rate, try reducing the magnetic divisor. A value of 256 is suitable to maintain 100 Hz sampling frequency.

The frequency at which the Colibri delivers data is determined by the *External trigger* setting, which indicates if the IMU is triggered from an external source or by the *Frequency* setting. Not all Colibri's are delivered with the possibility to be externally triggered, please contact info@trivisio.com for details and availability.

Some Colibri's also come with the ability to emit a trigger signal (again contact info@trivisio.com for details and availability). For these the *Trigger output divisor* determines how often a trigger signal should be emitted. If set to 0 no trigger signal is output, and otherwise the output frequency is the sensor frequency divided by the divisor.

In addition, the option *ASCII* turns on output suitable for terminal debugging can be activated, the *AutoStart* determines if the sensor starts automatically to output measurements when plugged in, and the *RAW* mode can be used to get raw uncalibrated data from the IMU.

Turning on *Jitter reduction* gives a more stable orientation estimate, as explained in Section 3.5.

Note: the available graphical output depends on the settings of *ASCII* and *RAW*. The modes and the according enabled widgets are shown in the following table.

2. Start/stop button.

   The currently selected Colibri can be started and stopped using this button.

3. Rescan button.

   Use the rescan button to search the system for connected sensors.

4. Save configuration button.

   Normally, the selected sensor configuration is downloaded (but not saved to the Colibri) as it is started and will be reset once the power is lost. Use this button to save the configuration to the Colibri's permanent memory.

5. Euler angle orientation.

   This describes the rotation of the Colibri as a sequence of three rotations, yaw (around the sensors positive z-axis), pitch (around the sensors negative y-axis), and finally roll (around the positive x-axis). Note that boresighting changes the sensors axis.

6. OpenGL® representation of the sensor orientation.

   The OpenGL® textures can be enabled or disabled by the menubar item **Configuration**.

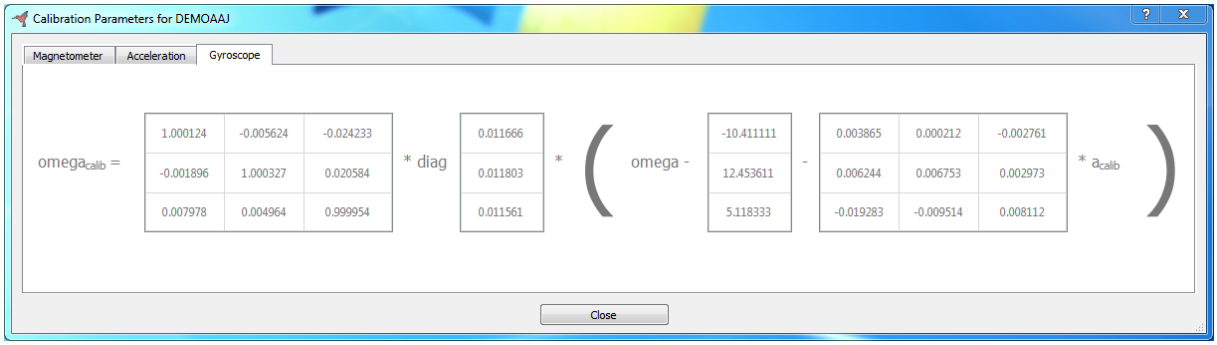
7. Graph of the magnetometer measurements

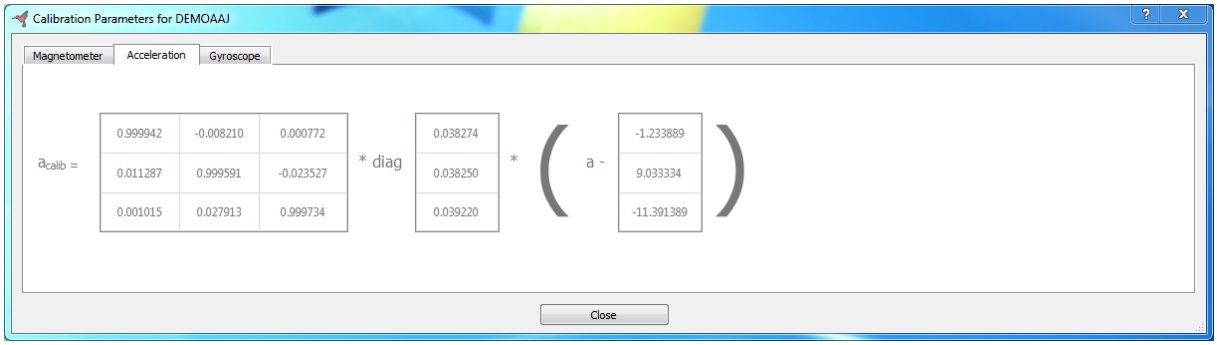
8. Graph of the accelerometer measurements.

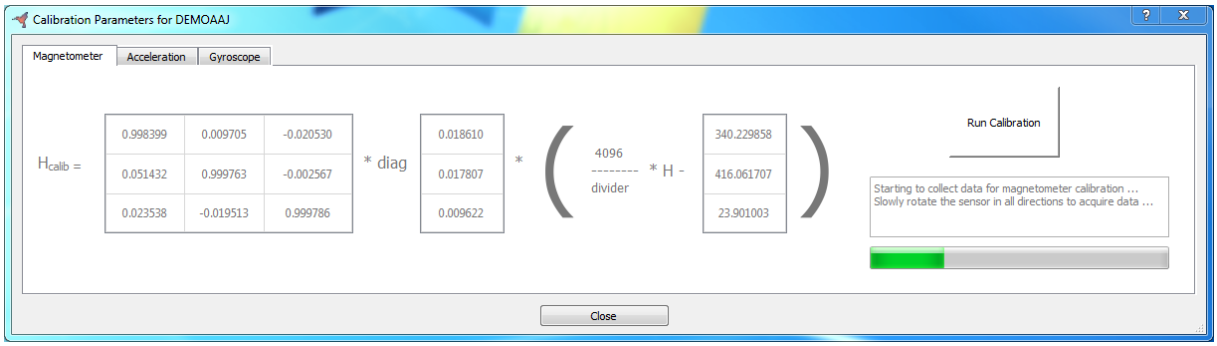
9. Graph of the gyroscope measurements.

3.3 Calibration parameters

To view the calibration parameters used to calibrate the IMU, use the **Calibration Parameters** menu item in the **Device** menu. The dialog looks as follows:







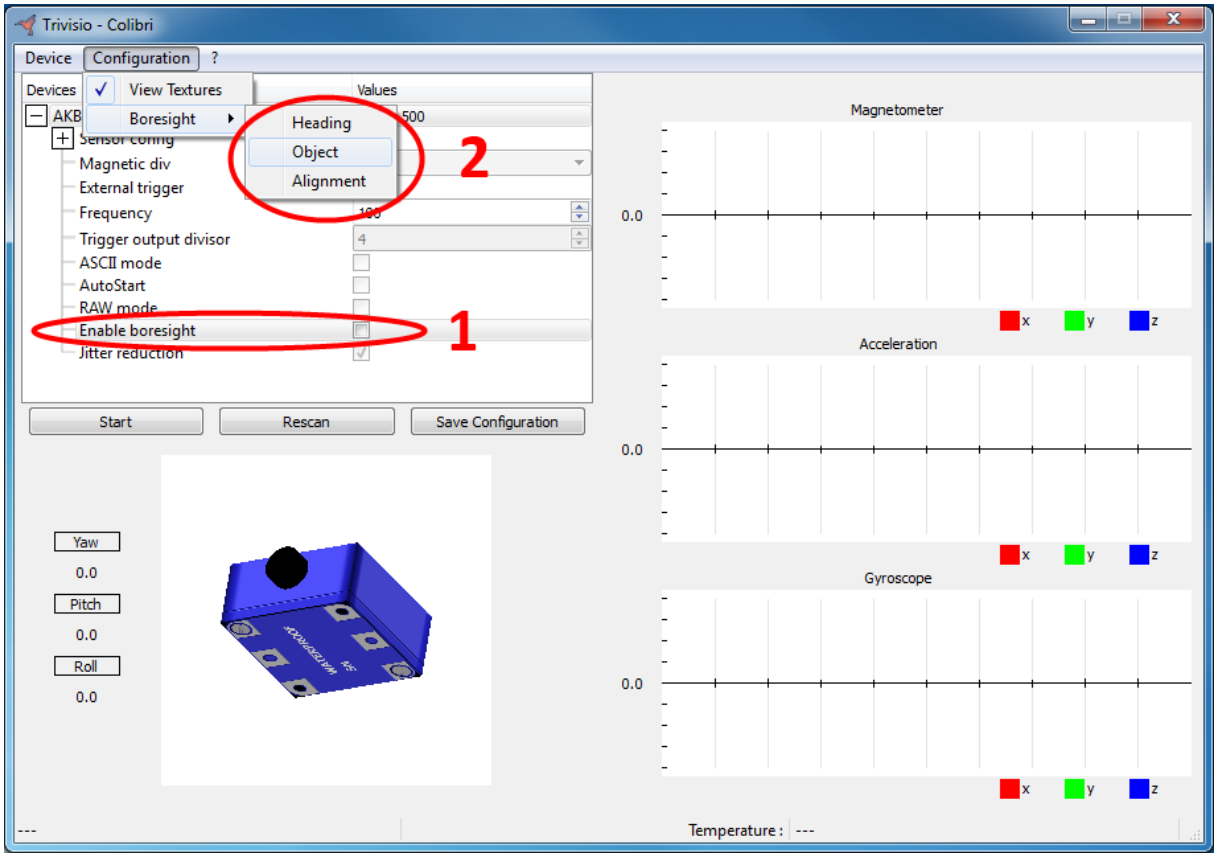
The Colibri sensors are factory calibrated and tested before being shipped. However, the magnetic calibration is very sensitive to the environment in which the sensor is used. This means that to obtain best possible orientation estimates, the Colibri should have its magnetometers recalibrated after it has been mounted in the way it is intended to be used. Not doing so will result in degraded orientation estimates.

To recalibrate the magnetometers, press the **Run Calibration** button in the view for the magnetometer calibration parameters. Then turn the sensor around all its axis in a smooth motions. If the calibration is successful, the parameters are downloaded to the IMU, and if not the calibration is simply left unchanged.

3.4 Boresighting

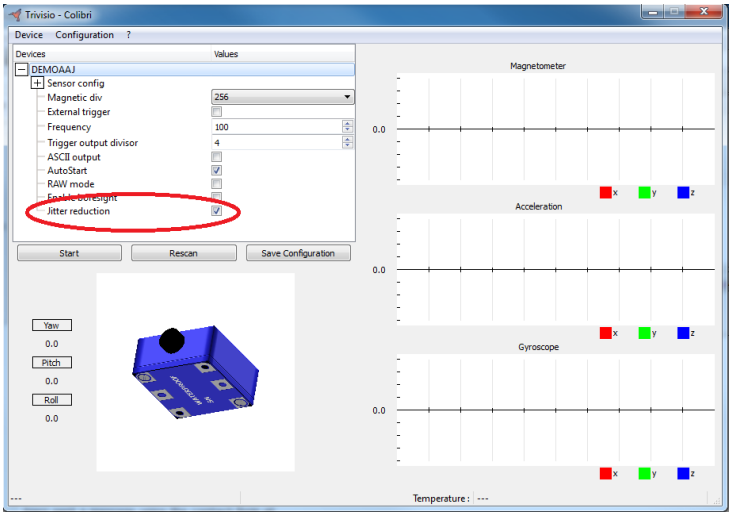
Boresighting can enabled and disabled in the tree view of the sensor (see **1** in the image below). When turned on this way, the Colibri will be boresighted according to the parameters saved in the sensor. The menu items in the **Configuration→Boresight** menu allows the user to update these parameters (see **2** in the figure below). Three different types of boresighting are possible:

- *Heading reset*, global frame is aligned with the current sensor x-axis (set yaw = 0). The vertical direction remains unchanged.
- *Object reset*, which changes the sensor coordinate system to align its z-axis with gravity. The heading is unchanged.
- *Alignment reset*, which combines heading and object reset.



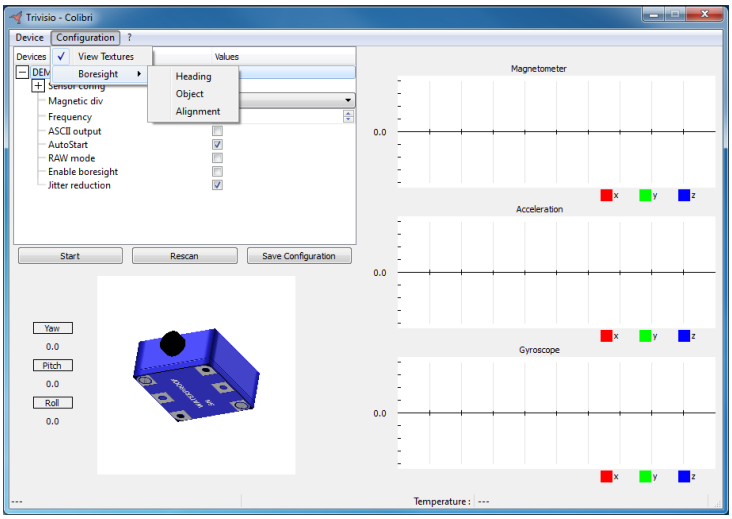
3.5 Jitter Reduction

In settings where it is more important to have a smooth orientation estimate than high precision (especially when the Colibri is nearly stationary) jitter reduction should be turned on. This can be done with the checkbox in the expanded tree view of the sensor. If you check it, jitter reduction is enabled otherwise not.



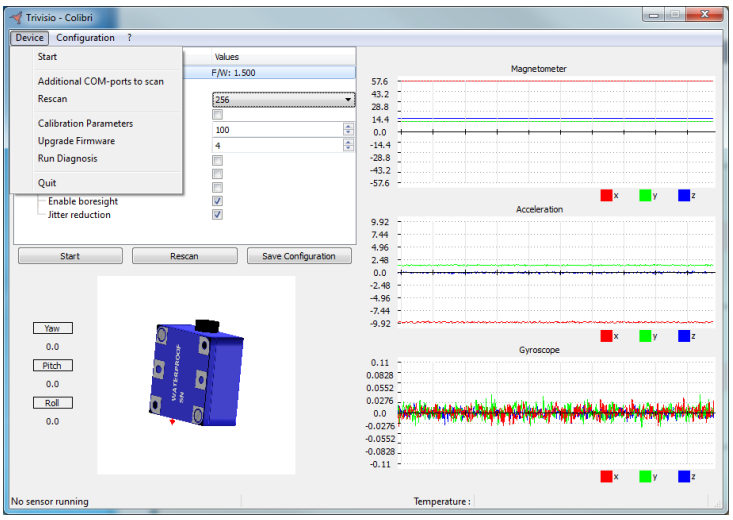
3.6 Additional functionality

3.6.1 Disable drawing of textures

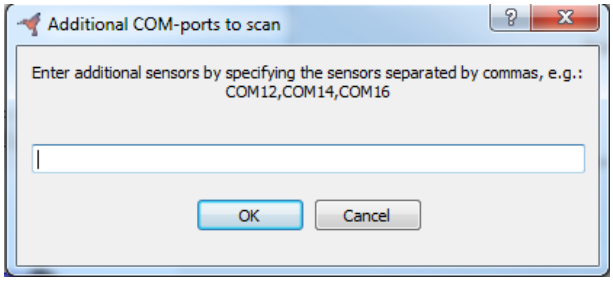


The menu bar contains the **Configuration** menu entry (see image above). If the graphics card is not powerful enough to render the textures, the textures can be switched off to improve the performance.

3.6.2 Additional COM ports to scan



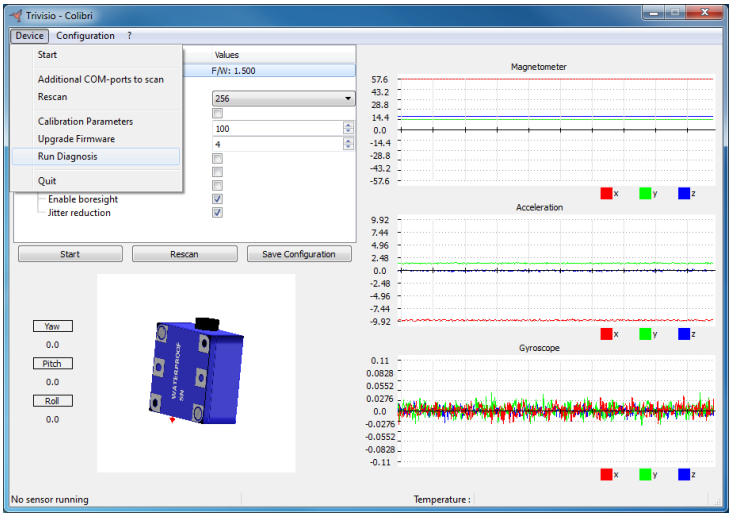
The Trivisio GUI does its best to find all available sensors, but it sometimes fails due to system specifics. One such reason is having a sensor connected via RS232. In this cases it is possible to manually specify additional COM ports or devices to be scanned for Colibris. To do so, select the **Additional COM ports to scan** from the **Device** menu. It is then possible to enter additional COM ports and devices in the dialog that opens up (see below). **Note:** In order to scan for the new ports and have the devices in the list, you **must** press the **Rescan** button!



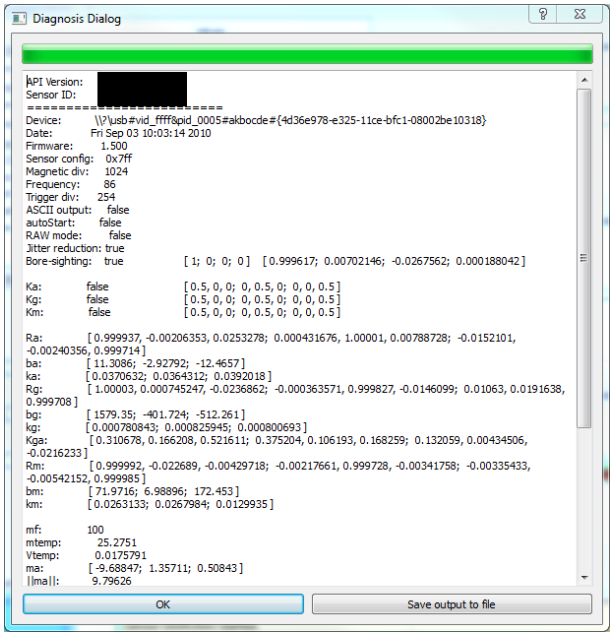
3.6.3 Sensor Diagnosis

The GUI comes with the possibility to derive diagnostic sensor information to simplify fault analysis. The diagnostic information contains information about all settings stored in the Colibri, as well as the result from a few seconds of collected data. This provides valuable information about the IMU such as bias and noise levels in the current setting. Please, always provide this information with support requests.

To obtain the diagnostic information, select **Run Diagnosis** from the **Device** menu and keep the sensor as stationary as possible. The best information is obtained if the sensor is diagnosed in the same environment as it is used.

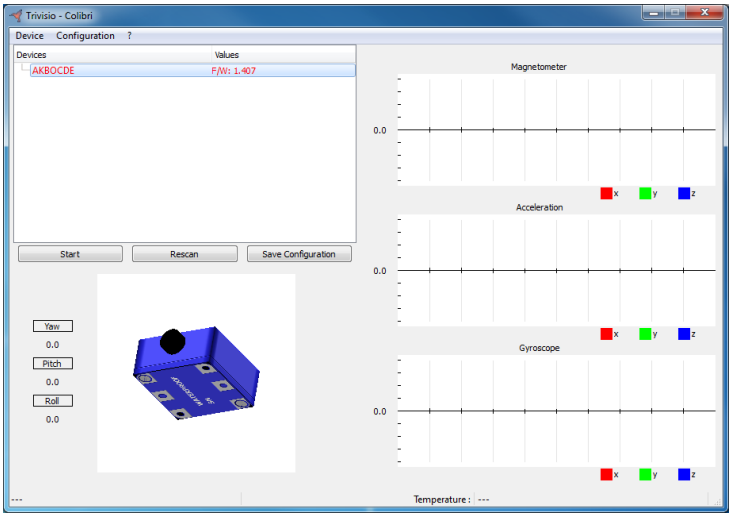


The result dialog can be seen below. It is possible to save the output to a text file which can easily be attach to support requests.

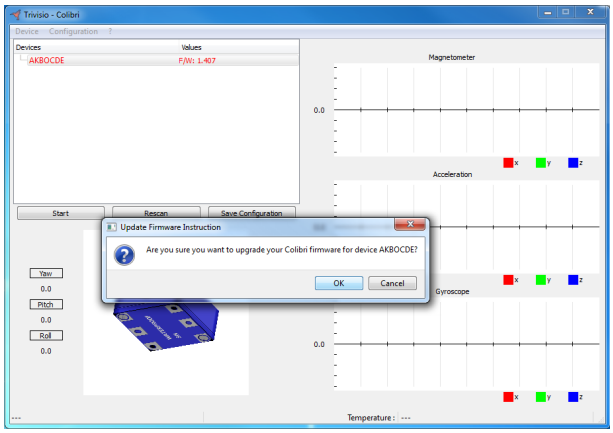


3.6.4 Firmware Upgrade

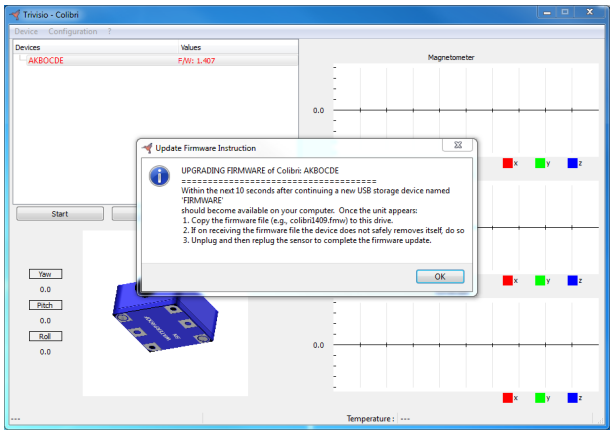
If your sensor is marked as red in the device list, the firmware of the sensor must be upgraded.



To upgrade the firmware, select the sensor in the list and select **Upgrade Firmware** from the **Device** menu. You are next asked, if you are sure to upgrade your firmware:



By pressing **OK** a new dialog will open with instructions on how to upgrade the firmware. Please follow the instructions.


Update Firmware Instruction
UPGRADING FIRMWARE of Colibri: AKBOCDE
Within the next 10 seconds after continuing a new USB storage device named FIRMWARE
should become available on your computer. Once the unit appears:
1. Copy the firmware file (e.g., colibri1409.fmw) to this drive.
2. If on receiving the firmware file the device does not safely removes itself, do so
3. Unplug and then replug the sensor to complete the firmware update.
OK

Within 10 seconds after initiating the firmware upgrade procedure a new USB mass storage device named 'FIRMWARE' should become available on your computer. Once the unit appears:

1. Copy the firmware file (e.g., colibri1500.fmw) to this drive. The installer comes with a firmware version that should be compatible with the SDK version. It can be found in the `firmware` folder of the installed program. The latest firmware can also be acquired from info@trivisio.com.

2. When receiving a new firmware file the USB mass storage should safely remove itself. If this does not happen, for example due to aggressive caching, safely remove it manually to speed up the procedure.

3. Unplug and then replug the sensor to complete the firmware update. Check the sensor entry in the GUI to see that the upgrade was successful.

Chapter 4

SDK

Two example programs are provided with the SDK. The first one is a console application named `ColibriTestC` (`testc.c`) and the second one is a GLUT® application. The Windows SDK also comes with an example on how to communicate with the sensor using C#.

The two applications can be found in the installation directory in the `example` directory. The Windows installation contains Visual Studio project for the two examples and a solution in the installation directory. The Linux installation includes the makefile `Makefile.linux`.

The required header files are in your installation directory in the `include` folder. The libraries are found in the `lib` or `bin` directory.

4.1 The `test.c` example

This section describes the `test.c` application.

The `TrivisioColibri.h` header is needed to use the Colibri API.

```
1  #include "TrivisioColibri.h"
2
3  #include <stdio.h>
4  #ifdef WIN32
5  # include <windows.h>
6  #else
7  # include <unistd.h>
8  #endif
9
10 #define M_PI 3.14159265358979323846
```

This simple example contains just a `main` function. In it the Colibri sensors are scanned and the measurements are printed to the console. In addition the methods for getting and setting the configuration will be shown.

```
12 int main()
13 {
14   struct TrivisioSensor sensorList[10];
15   int sensorCount = colibriGetDeviceList(sensorList, 10);
```

The available devices can be detected by calling the `colibriGetDeviceList` function. The function returns the number of available sensors, and it takes two parameters; an array of sensors to be populated and the size of this array.

```
17   void* imu = colibriCreate(100);
```

This function creates the Colibri devices, whereas the parameter is the length of the buffer internal buffer.

```
18   struct ColibriConfig conf;
19   char ID[8];
20
21   /* Diagonal matrices with diagonal element .68 yields approx 20Hz
22      bandwidth @ 100Hz */
23   float Ka[9] = { 0.68f,    0.00f,    0.00f,
24                   0.00f,    0.68f,    0.00f,
25                   0.00f,    0.00f,    0.68f };
26   float Kg[9] = { 0.68f,    0.00f,    0.00f,
27                   0.00f,    0.68f,    0.00f,
28                   0.00f,    0.00f,    0.68f };
29
30   struct TrivisioIMUData data;
31   unsigned oldt = 0;
32   int i;
```

In this block, custom pre-filtering parameters are generated and will later on be transmitted to the sensor. Using diagonal matrices with 0.68 as diagonal element yields independent filtering of each data channel with a bandwidth of approximately 20 Hz. Some variable needed later are also declared.

```
34   printf("Number of Colibris found: %d\n", sensorCount);
35   if (sensorCount<0)
36     sensorCount = 10;
37   for (i=0; i<sensorCount; ++i)
38     printf("%s:\t %s (FW %d.%d)\n", sensorList[i].dev, sensorList[i].ID,
39                                     sensorList[i].FWver, sensorList[i].FWsubver);
40   printf("\n\n");
41
42   if (sensorCount<1) {
43     fprintf(stderr, "No Colibri sensors found\n");
44     return 0;
45   }
```

This block prints the available sensors to the console. The printing shows

- the device (`sensorList[i].dev`), e.g. COM1,
- the id of the sensor (`sensorList[i].ID`),
- the firmware version of the sensor (`sensorList[i].FWver`),
- and the firmware sub verion number of the sensor (`sensorList[i].FWsubver`).

```
46   if (colibriOpen(imu, 0, 0) < 0) {
47     fprintf(stderr, "Error while trying to access Colibri\n");
48     return -1;
49   }
```

Try to open a sensor by calling `colibriOpen`. The parameters are the `imu`, which was created earlier, a predefined configuration of the sensor, and a device port. When successfully opening a communication channel to the sensor, the function returns 0.

```
51    colibriGetConfig(imu, &conf);
52    conf.raw = 0;
53    conf.freq = 100;
54    conf.sensor = ALL;
55    conf.ascii = 0;
56    colibriSetConfig(imu, &conf);
```

Retrieve the current configuration of the acquired sensor, and set raw and ascii mode, frequency, and sensor configuration to the desired values. The configuration is then written back to the sensor using `colibriSetConfig` to take effect.

```
58    colibriSetKa(imu, Ka);
59    colibriSetKaStatus(imu, 1);
60    colibriSetKg(imu, Kg);
61    colibriSetKgStatus(imu, 1);
62    colibriSetJitterStatus(imu, 1);
```

Next preprocessing of the accelerometer and gyroscope data is activated, as well as jitter reduction.

```
64    printf("Colibri IMU\n");
65    colibriGetID(imu, ID);
66    printf("Device ID:          %s\n", ID);
67    printf("Sensor config:      %d\n", conf.sensor);
68    printf("Magnetic div:       %d\n", (unsigned)conf.magDiv);
69    printf("Frequency:          %d\n", conf.freq);
70    printf("ASCII output:       %d\n", conf.ascii);
71    printf("autoStart:          %d\n", conf.autoStart);
72    printf("RAW mode:           %d\n", conf.raw);
73    printf("Jitter reduction: %d\n", colibriGetJitterStatus(imu));
```

And the sensor settings are printed.

```
76    colibriStart(imu);
```

Start the colibri by calling the function `colibriStart`.

```
77    for (;;) {
78      colibriGetData(imu, &data);
79      if (data.t > oldt) {
80        float eul[3];
81        printf("Time: %6.2f\t", data.t*1e-4);
82        printf("Temp: %6.2f\t", data.temp);
83        printf("Acc: %6.2f, %6.2f, %6.2f\t", data.acc_x, data.acc_y, data.acc_z);
84        printf("Gyr: %6.2f, %6.2f, %6.2f\t", data.gyr_x, data.gyr_y, data.gyr_z);
85        printf("Mag: %6.2f, %6.2f, %6.2f\t", data.mag_x, data.mag_y, data.mag_z);
86        printf("Quat: %6.2f, %6.2f, %6.2f, %6.2f\t",
87               data.q_w, data.q_x, data.q_y, data.q_z);
88        colibriEulerOri(&data, eul);
89        printf("Euler: %10.4f, %10.4f, %10.4f\n",
90               180/M_PI*eul[0], 180/M_PI*eul[1], 180/M_PI*eul[2]);
91        oldt = data.t;
92      }
93  #ifdef WIN32
94      Sleep(2);
95  #else
96      usleep(2000);
97  #endif
98    }
```

The data will be read out of the sensor by the function `colibriGetData`. The Euler orientation can be obtained by calling `colibriEulerOri`.

```
100    colibriStop(imu);
101    colibriClose(imu);
```

The sensor is stopped by calling `colibriStop` and closed by `colibriClose`.

4.2 The `orientation.cpp` example

The `orientation` program is a short illustration of how to use the Colibri in a graphical application. Since the `testc.c` has already been explained in detail, only the new functions will be explained.

Key shortcuts:

'c' toggles visualisation of cube

'j' toggles jitter reduction

'n' toggles visualisation of Euler angles as numbers

'q' quits the program

The function `colibriSetJitterStatus(imu, status)`, where `status` is either `true` or `false`, turns jitter reduction on or off.

Three different types of boresighting are provided (see Section 3.4 for details):

'h' heading reset, (assume yaw = 0 when 'h' is pressed)

'o' object reset, align the up axis of the sensor with gravity, keeping the yaw as it is

'a' alignment reset, (current pose is matched with the nominal pose, h+o)

'r' reset alignment (undo any previous adjustment, that is turn of boresighting)

Boresighting is activated with `colibriSetBoresight(imu, 1)` and deactivated with `colibriSetBoresight(imu, 0)`. To boresight, use `colibriBoresight(imu, type)`, where type should be one of: `COLIBRI_HEADING_RESET` (for heading rest), `COLIBRI_OBJECT_RESET` (for object reset), and `COLIBRI_ALIGNMENT_RESET` (for alignment reset).

4.3 Simple Import mechanism for data profiling

In order to log the output to a file, use `ColibriTestC.exe` (Windows) or `ColibriTestC` (Linux) to pipe the data to a file as as follows:

Windows: `ColibriTestC.exe > YOUR_FILE_NAME.txt`

Linux: `ColibriTestC > YOUR_FILE_NAME.txt`

The file can be imported with a spread sheet application, e.g., Ms Excel or OpenOffice. To correctly extract the data, use ',', ' ', and Tab as delimiters, and merge adjacent delimiters.